SYTI 5

Bewertetes Übungsbeispiel

Name: Paul Scheidl

Punkte: 37 + 6

Viel Erfolg!

Humidity Control: Im Detail gilt es folgende Anforderungen zu realisieren: Das Embedded System verfügt über 2 Feuchtigkeits-Sensoren. Die aktuellen Messwerte sind auf dem LCD darzustellen. In Abhängigkeit des zu ermittelnden Mittelwertes gilt es Warn-LEDs zu schalten (Details siehe unten). Messungen sollen im 1- und 4-Sekunden-Intervall möglich sein. Das Setzen des aktuellen Intervalls hat via USART zu erfolgen. Das System verfügt außerdem über 3 Tasten, die wie folgt zu implementieren sind: Bei Betätigung der Taste 1 soll ausschließlich Sensor 1 bei den Warn-LEDs bzw. der LCD-Ausgabe berücksichtigt werden. Analog ist Taste 2 zu implementieren. Bei Taste 3 wiederum beide.

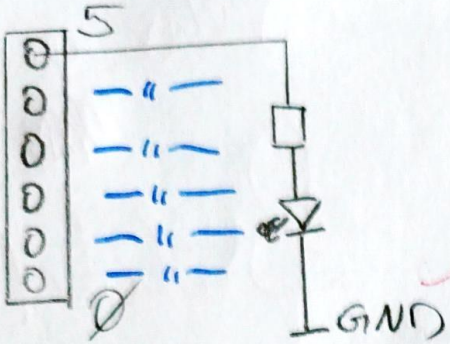
Es gilt:

- Taste 1: Sensor 1 aktiv
- Taste 2: Sensor 2 aktiv
- Taste 3: Sensoren 1 und 2 aktiv
- Feuchtigkeitssensoren
  - Zu Berechnung des eigentlichen RH-Wertes gilt die es die Library rh.h zu verwenden, die folgende Funktion bereitstellt:
  - `uint_8 calcHumi(uint_16 adcVal);`
  - Dieser ist der aktuelle ADCW-Wert zu übergeben, als Return-Wert ist ein Integer zw. 0 und 100 zu erwarten.
- LCD
  - Ausgabe 2-zeilig:
    - RH S1: 75%
    - RH S2: 73%

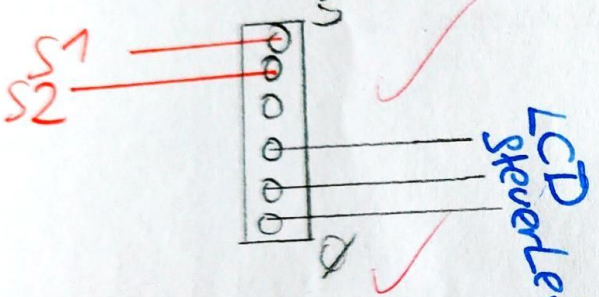
    Logal nur 1 Wert
- Warn-LEDs
  - RH > 80% => LED 3, 4 und 5 (rot)
  - RH > 50% => LED 1 und 2 (orange)
  - sonst => LED 0 (grün)
- USART – Timer-Intervall setzen
  - Zeichen 1: 1 Sekunde
  - Zeichen 4: 4 Sekunden
- Intervalle: Timer-basiert
- Hinweise:
  - Führen Sie den/die Port(s) bzw. dessen Pins an, an welche Sie das LCD schalten. Detaillierten **Schaltplan** (inkl. Tasten) zeichnen!
  - Es ist von entprellten Tasten auszugehen.
  - Falls notwendig, treffen und dokumentieren Sie plausibel Annahmen!

PORT B

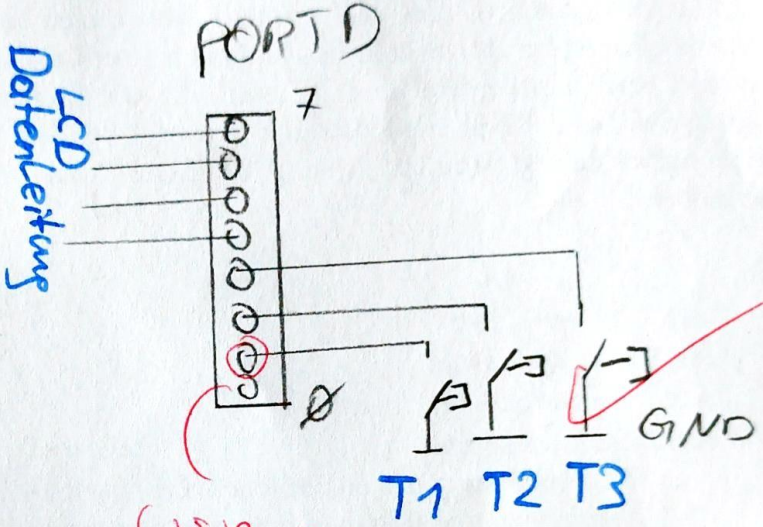
5,4,3 => ROT
2,1 => GELB
0 => GRÜN



PORT C



PORT D



USART!!!

```c
/*
 * BWUbengsbseipiel.c
 *
 * Created: 19.01.2023 10:42:35
 * Author : p.scheidl
 */
#define F_CPU 16000000
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdio.h>
#include "lcd.h"

void innit();
void Muxchanger(uint8_t);

//string buffer
char buffer[16]="";

//1=Sensor 1; 2= Sensor 2; 3= Beide sensoren
volatile int sonsorselct = 1;
volatile int needsecondMessung = 0;

//Timeselect
volatile int timeselect = 1;

//RH Wert
volatile unsigned int wert = 0;
volatile unsigned int zwwert = 0;

int main(void)
{
    innit();
    /* Replace with your application code */
    while (1)
    {
    }
}

void innit(){
    //LCD Init
    lcd_init(LCD_DISP_ON);

    //LEDs als ausgang
    DDRB = 0xff;

    //Tasten intern Pull Up und Gruppen Interrupt
    PORTD |= (1<<PORTD1)|(1<<PORTD2)|(1<<PORTD3);
    PCICR |= (1<<PCIE2);
    PCMSK2 |= (1<<PCINT17) |(1<<PCINT18) |(1<<PCINT19);

    //UART Konifg Ich will empfangen RXEN0 und IE für Interupt enable
    UBRR0 = 103;
    UCSR0B |= (1<<RXEN0) |(1<<RXCIE0);
    //8 Databits
    UCSR0C |= (1<<UCSZ00) |(1<<UCSZ00);

    //TimerCOnfig (1024 PRescaler)
    TCCR1B |= (1<<CS12) | (1<<CS10);
    //RECHNUNG: 1024 * 65536 / 16000000 = 4,16s  (Controller hat Frequen 16Mhz = 1/16*10^6 s => ein Zählschritt ohne Pres
    //1024 Prescaler = man zählt nur all 1024 schritte und 65536 16Bit timer kann diese anzahl an schritten gehen
    // Ein Schritt 0,000 064s
    // 1 Sekunde = 15625 Schritte
    // 4 Sekunden = 62500 Schritte
    //Vorlanden: 65536-62500 = 3036; 65536-15625 = 49911
    TCNT1 = 49911;
    TIMSK1 |= (1<< TOIE1);

    //ADC Konfig: REF für Referenz Spannung 5V; ADPS = Presacler auf Maximum; IE Inerrupt enabable
    //S1: MUX 101 s2: Mux 100
    ADMUX |= (1<<REFS0) | (1<<MUX2);
    ADCSRA |= (1<<ADEN) | (1<<ADIE)|(1<<ADPS2) |(1<<ADPS1) |(1<<ADPS0);

    sei();
}
ISR(ADC_vect){
    if(needsecondMessung == 1){
        zwwert = calcHumi(ADCW);
        needsecondMessung =0;
        ADCSRA |= (1<<ADSC);
    }
    else{
```

Super Arbeit, Paul!

Du Mux neu setzen!

Vor der 2. Messung.

```
        wert = calcHumi(ADCW);
        if(zwwert != 0){
            wert = (wert+zwwert)/2;
            zwwert=0;
        }
        lcd_clrscr();
        sprintf(buffer, "RH: %u%%", wert);
        lcd_puts(buffer);

        if(wert > 80){
            PORTB = 0x00;
            PORTB |= (1<<PORTB3) |(1<<PORTB4) |(1<<PORTB5);
        }
        else if(wert > 50)
        {
            PORTB = 0x00;
            PORTB |= (1<<PORTB1) |(1<<PORTB2);
        }
        else{
            PORTB = 0x00;
            PORTB |= (1<<PORTB0);
        }
    }
}
ISR(TIMER1_OVF_vect){
    //Timer für Mesung deaktivieren
    TIMSK1 &= ~(1<< TOIE1);

    //Messvorgang starten, voher überrüfen welcher Sensor, wenn beide needsecond
    if(sonsorselct == 1){
        Muxchanger(4);
    }
    else if(sonsorselct == 2){
        Muxchanger(5);
    }
    else if(sonsorselct == 3){
        Muxchanger(4);
        needsecondMessung = 1;
    }
    ADCSRA |= (1<<ADSC);

    //Nächsten Timer mit Richtigen Intervall starten
    if(timeselect == 1){
        TCNT1 = 49911;
    }
    else if(timeselect == 4){
        TCNT1 = 3036;
    }
    TIMSK1 |= (1<< TOIE1);
}

void Muxchanger(uint8_t ab){
    //S1: MUX 101 s2: Mux 100
    if(ab==5){
        ADMUX |= (1<<MUX0);
    }
    else if(ab==4){
        ADMUX &= ~(1<<MUX0);
    }
}

ISR(USART_RX_vect){
    //1 oder 4
    timeselect= UDR0;
}

ISR(PCINT2_vect){
    //Überprüfe welcher Button gedrückt
    if(!(PIND & (1<<PORTD1))){
        sonsorselct = 1;
    }
    else if(!(PIND & (1<<PORTD2))){
        sonsorselct = 2;
    }
    else if(!(PIND & (1<<PORTD3))){
        sonsorselct = 3;
    }
}
```